L'objectif de ces exercices est de travailler les notions du chapitre 9 : les strings

Exercice 9.1

Voyelles

Complétez le programme en y ajoutant une fonction `compterVoyelles()` qui prend un mot en paramètre et renvoie son nombre de voyelles.

Une voyelle en majuscules reste une voyelle... A vous d'en tenir compte. En revanche, vous n'êtes pas obligé de gérer les accents

```
// Ajoutez votre code ici

console.log(compterVoyelles("RadAr")); // 2
console.log(compterVoyelles("Tic et Tac")); // 3
console.log(compterVoyelles("Oasis Oasis Oh")); // 7
```

Résultat attendu :

```
2
3
7
```

Exercice 9.2

Palindrome

Complétez le programme pour définir la fonction est Palindrome() qui renvoie vrai ou faux selon que le mot soit un palindrome ou non.

Il existe deux techniques pour construire le mot inversé :

- Parcourir le mot initial lettre à lettre en ajoutant chaque lettre au début (et non à la fin) du mot inversé.
- Parcourir le mot initial lettre à lettre, mais à l'envers (de la fin vers le début).

La vérification ne doit pas tenir compte des distinctions entre majuscules et minuscules : "RADAR" est un palindrome, "Radar" aussi.

```
// Ajoutez votre code ici

console.log(estPalindrome("RadAr")); // true
console.log(estPalindrome("KAYAk")); // true
console.log(estPalindrome("Bora-Bora")); // false
```

Résultat attendu :

```
true
true
false
```

Exercice 9.3

1337 5|*34|<

Le leet speak est un système d'écriture où certains caractères sont remplacés par d'autres afin de produire un résultat différent mais visuellement proche. Il est ou était souvent utilisé dans certaines communautés hackers et gamers.

Il existe de nombreuses variantes de l'alphabet leet. Je vous propose d'utiliser au minimum le suivant, que vous pourrez enrichir si vous le souhaitez.

Lettre	Equivalent leet
a	4
b	8
e	3
l	1
o	0
s	5

La conversion doit fonctionner indifféremment pour une lettre minuscule ou majuscule.

Complétez le programme en définissant la fonction `convertirMotLeet()` qui prend en paramètre un mot et renvoie son équivalent leet.

Afin d'alléger le code de la fonction `convertirMotLeet()`, vous pouvez créer une autre fonction `convertirLettreLeet()` qui prend en paramètre une lettre et renvoie son équivalent leet. Cette fonction sera appelée pour chaque lettre du mot initial.

```
// Ajoutez votre code ici

console.log(convertirMotLeet("Hello World!")); // "H3110 W0r1d!"
console.log(convertirMotLeet("Noob")); // "N008"
console.log(convertirMotLeet("Hacker")); // "H4ck3r"
```

Résultat attendu :

```
H3110 W0r1d!
N008
H4ck3r
```

Exercice supplémentaire 9.4

Truncate

Ecrivez une fonction JavaScript pour tronquer une chaîne si elle est plus longue que le nombre de caractères spécifié.

Les chaînes tronquées se termineront par une séquence d'ellipse traduisible ("…") (par défaut) ou des caractères spécifiés

```
document.write(text_truncate('We are doing JS string
exercises.',15,'!!'))
```

Exercice supplémentaire 9.5

Humanize

Ecrivez une fonction JavaScript sur un nombre humanisé (Formate un nombre en une chaîne lisible par l'homme.) Avec le suffixe correct tel que 1er, 2ème, 3ème ou 4ème

```
document.write (humanize_format (1));
document.write (humanize_format (402));
document.write (humanize_format (11));
"1er"
"402ème"
"11ème"
```

2ème version pour la notation anglaise 1st, 2nd, 3rd or 4th.

```
document.write (humanize_format(301));
document.write (humanize_format(402));
document.write (humanize_format(303));
document.write (humanize_format(104));
"1st"
"402nd"
"303rd"
"104th"
```

On peut spécialiser la fonction avec un 3ème paramètre. fr, en etc…

```
document.write (humanize_format(301,"fr"));
301ème
document.write (humanize_format(301,"en"));
301st
```

Exercice supplémentaire 9.6

Email

Pour les plus avancés !

Créer une fonction qui contrôle une chaîne de caractères est une adresse E-mail valide.

Voici un set de test :

```
@.-->false
b@c.dd-->true
a.b@c-->false
aa@bb.cc-->true
aa.bb@c.fr-->true
```